IN THE SPECIFICATION

Please amend the paragraph at page 4 lines 6 – 18 as follows:

The shared file system is purely centralized. As a result, it provides only one context – itself. Pure centralization thus sacrifices context in favor of a uniform and shared file structure. But how can two users work on the same document or stage a review cycle? Typically, users resort to e-mail. A single, shared structure is incapable of supporting the requisite context of interaction among workers in a collaborative setting. For example, consider what occurs if a team creates two shared spaces, and accessed by different members. By creating multiple information spaces, the team has simply fragmented, or disintegrated the information product. In gaining collaborative "context" (e.g., ~~for example,~~ a private shared information space), the team typically fragments the collaborative information product and loses continuity. Finally, allowing users to simultaneously edit to a file does not in itself provide context, since there remains only one (albeit shared) information context (e-mail remains the only medium for contextual collaboration).

Please amend the paragraph at page 10 lines 8 – 9 as follows:

Figure 23 is a flowchart of an embodiment showing steps of creating derived <u>access</u> for a group of recipients;

Please amend the paragraph at page 11 lines 13 – 23 as follows:

The server 105 and associated database 110 may together persist and manage information and data. The server 105 and database 110 may exist in other variations, as one skilled in the art would recognize. The server may support different network applications (e.g., ~~for example,~~ word processor, enterprise applications, data base applications, or the like). The server 105/135 typically contains the set of data and information accessed by the client. The clients 115/130/135 typically provide the

interface through which users (knowledge workers) access a network application. The client may also store and execute software belonging to the network application (e.g., ~~for example,~~ thick client applications, such as word processors) or may also provide the application through a thin client interface (e.g., for ~~example,~~ the internet browser, terminal services). Client firmware include the PC or thin web client.

Please amend the paragraph at page 11 lines 24 – 32 as follows:

The Internet 125 may connect remote clients and peers to the network 120. The remote client may access the network 120 through security mechanisms, including a firewall and virtual private network (VPN). The peer 135 is typically both a server and a client, including the database 140, a server / client 135 within a ~~in a~~ single unit (e.g., a laptop, having a database stored on its local hard drive, the server stored and executed locally, and network applications stored and executed locally). Servers 105 and 135 are typically called "peers" when they are connected by Internet 125 and network 120. As peers, servers 105 and 135 may jointly manage the "data environment".

Please amend the paragraph at page 12 lines 4 – 10 as follows:

The CDM 145 may logically exist across one or more physical data ~~repositories.~~ repositories (may physically reside but is logically one thing). The CDM may be present and pervasive throughout the entire network. The CDM is pervasive as servers execute the method of information, pushing data across the network (e.g., into user view), making it pervasive. The data is complete and self describing within a continuum. Encompassing the entire network and across users, networks, system data, time.

Please amend the paragraph at page 13 line 22 to page 14 line 4 as follows:

The process of building up a form typically involves building a new form out of existing ones. At the beginning of the process, the user typically combines primitive

types that may be built into the system (e.g., for example, integer or string). The system database typically has tables corresponding to each element of the "System:Forms", "System:Elements", "System:Fields" (and other elements of the form), since each instance is typically itself an entity. Therefore, creating a new form typically results in a new instance of a Form entity (typically stored as a record in the "System:Forms" table), as well as additional Element entities that may be contained by the Form (typically stored as records in the "System:Elements" table), as well as additional Field entities that may be contained by the Form (typically stored as records in the "System:Fields" table), as well as other entities that may be contained by the form structure (e.g., ~~for example,~~ including the content groupings Choice and Sequence), as well as containment relationships typically creating the complex data structure of a form (typically stored as records in the "System:Containment" table), as well as any base relationships that may connect the form entity with base forms or form elements with base elements (typically stored as a record in the "System:Base" table) , and so on.

Please amend the paragraph at page 14 lines 5 – 17 as follows:

Creating a new form typically also prompts the system to create the database table that may contain entity instances of the form. A database table typically embodies the entity instance of a form. The database table typically comprises a set of columns corresponding to each field of the form. The database table is typically automatically created by the system when the new form is defined (e.g., ~~for example,~~ when the complex data structure of a form and elements is created). The table's name typically adopts the qualified name of the form (e.g., ~~for example,~~ "System:FormName"). An entity instance of the form may then be contained as a record in the table corresponding to the form. Elements of an the instance of a form instance are typically connected to the instance of a form by containment relationships. For example, a containment relationship may connect a document instance (whose form is document) with its nested subdocument(s) (which may be defined as element(s) of the document's form).

Please amend the paragraph at page 14 lines 18 – 25 as follows:

[[|]]The CDM comprises a graph/network of entities and relationships. The entity may be an anchor of reference, or relationship target (the relationship may be implemented as an entity subtype). The entity may have a set of properties/fields/attributes. The relationship may be a k-tuple, wherein each member may be a reference to a target entity or a literal value. The meaning of an entity may typically be defined by relationships connecting it with other entities. To this extent, as an entity may be defined vis-à-vis other entities, the relationships ~~defines~~ define both meaning and context of the entity.

Please amend the paragraph at page 15 lines 16 – 19 as follows:

Figure 3 is a logical block diagram showing components and elements of a CDM, generally denoted by reference numeral 300. The CDM encompasses, but is not limited to, system data 305, access controls 310, user data 315, time 320, and networks 325. The CDM unifies and integrates at least these various aspects.

Please amend the paragraph at page 17 lines 9 – 16 as follows:

The invention may reconcile physical data centralization & decentralization by including a method of identifying entities by a globally unique identifier. Unlike the World Wide Web (prior to the invention), the entity's unique identifier (UID) is not typically based on the data's physical location. Rather, the UID may be generated randomly with global uniqueness (for example, an embodiment may be the System.Guid.NewGuid() method included in Microsoft's .NET framework). Each entity may then be identified by its associated UID (an embodiment may include UID as a field in the "form" definition of the entity).

Please amend the paragraph at page 17 line 22 to page 18 line 4 as follows:

As a result, the invention may decouple the logical structure from the physical structure (i.e., location) of data. This allows the platform to automate, fully, the physical placement of data and its replication/synchronization. The manner of data placement / replication / synchronization may be optimized by the system on the basis of one or more:

(a) Security considerations: The system may hold data behind a firewall for purposes of security, or establish ~~one,~~ one trusted server as the access point for employees accessing data remotely.

(b) Efficiency considerations: The system may replicate and synchronize data across multiple platform data sites, to reduce latency and guarantee quality of service (QoS). The system may execute QoS constraints specified by the administrator or automatically determined by a user work load requirements.

(c) On-line, Off-line: The system may replicate data on sites which are expected to go off line. For example, the system may automatically replicate data onto a user's laptop (participating as a system site), so the user has access to files when disconnected from the network.

Please amend the paragraph at page 18 lines 11 – 19 as follows:

In one aspect, ~~The~~ the invention defines complexity as relational density in a data structure. A function of the system, according to the invention, may be to increase data complexity, on the basis that data manageability is a function of complexity. The process of complexity may have the effect of creating a "data continuum" in the CDM. The process of complexity may occur through the following operations, including linkage, expansion/granularization, and contraction/unification (contraction and expansion may considered symmetric elements of the process of complexity). These operations have the effect of increasing complexity and continuity of a given CDM, for example:

Please amend the paragraph at page 23 lines 15 – 25 as follows:

Figure 11C is a flowchart of an embodiment showing steps of parallel workflow, starting at 1155. At step 1160, user A may create derived access of entity E for users B and C with evolving and change access. At step 1165, either user A, B, or C may change entity E. At step 1170, the other two users may receive the change to entity E. Users B and C may typically receive each other's change since access may freely/bi-directionally ~~evolves~~ evolve in the tree containing access controls A, B, C. For example, when B makes a change, the change typically expands the access group of the derived access control (X), since no managed exclusion is held by access control B in the example. Access control C would then typically evolve to include the change since the 'evolving' bit of access control C is true in this example, thus demonstrating parallel work flow. At step 1175, the process ends.

Please amend the paragraph at page 23 line 26 to page 24 line 2 as follows:

Figure 11D is a flowchart of an embodiment showing steps of parallel workflow, starting at 1180. At step 1185, user A may create derived access of entity E for user B with manage / change access. At step 1190, user B may obtain exclusive/manage access. At step 1195, user B may create derived access of entity E for user A without manage/ change access. At step 1197, user B may change entity E. At step 1198, user A may receive B's change to entity E, but cannot change entity E~~. At step~~. In this example of parallel workflow, the recipient of derived access to an entity may obtain exclusive access of the entity, which may typically deny other users (e.g. user A) evolving access to the entity. But in this example, the user may provide read-only access back to user A (e.g. for purposes of review by user A), which may demonstrate parallel workflow. At step 1199, the process ends.

Please amend the paragraph at page 26 lines 3 – 12 as follows:

[[|]]An aspect of the invention includes a computer managed "complex data medium" (CDM), a network of information-bearing data. The invention dramatically increases productivity on the basis of the relational complexity that becomes the

informational component of data. Such "complexity" becomes the relational data "context" of data content. The invention maintains "relationships" (or "managed relationships") as the vehicle of complexity, and by extension, managed information. The invention serves to increase data complexity through interaction by multiple knowledge workers within a common information space. The invention fosters the cumulative effect of complex data, that is, compound value creation through a scalable information process.

Please amend the paragraph at page 27 lines 7 – 29 as follows:

Figure 15 is a flowchart of an embodiment showing steps of information delivery (a component of the process of information), starting at 1500. At step 1505, the user application may begin to "consume" an entity E. For example, the user may select, browse, or begin to edit the entity. The "consumer" may be considered the recipient of the process of information. At step 1510, the system may discover relationships attached to entity E. At step 1515, if the system discovers a relationship (including any new or existing relationship), it typically ~~proceed~~ proceeds to step 1520; otherwise, it typically returns to step 1410. For example, the system may discover a relationship (or relationships) associating a document consumed by the user with a comment/message, providing this comment/message as relational ~~information.~~ information ~~1510~~. At step 1525, the system may notify the user application that a new relationship is attached to entity E. The system typically provides this notification in real-time or near real-time. The system typically "pushes" information to the consumer (e.g., ~~for example,~~ the application/user). At step 1525, a parallel process (marked by a dotted line) may automatically return to step 1510, awaiting further relational information. At step 1530, the user application may automatically update the data view of entity 'E' to include the new relationship. At step 1535, the process ends. An actor may also "discover" relational information through a mechanism providing relationships connected to a given entity. The invention ~~include~~ includes this form of discovery as a component of the information process. For example, a user may instruct the system to discover relationships connected to a certain entity (and may apply parameters to the search). The user may receive back a set of relationships, for

example, showing every folder which contains the specified document through a containment relationship.

Please amend the paragraph at page 28 lines 10 – 18 as follows:

The invention may ~~provides~~ provide for a computer application (such as, for example, a word processor) to evolve from a unidirectional (one-way) to a bidirectional (two-way) data processor. The application may become a bidirectional medium since it is driven by both the user and the platform. Applications currently reach into a platform through the platform's application programming interface (API) (pull). However, the invention establishes the bidirectional channel, such that platform may also reach into the application, delivering information / direction (push). As such, the platform API may become a bi-directional data environment. In this way, the environment allows, in effect, the work of all users to reach into the environment of any single user.

Please amend the paragraph at page 28 line 19 to page 29 line 7 as follows:

An embodiment of the bidirectional interface may include the dynamic view. An embodiment of the dynamic view may be an XML projection ("XML View") of the CDM ~~that~~. The XML View is typically consumed by an application. The XML View may present a flat, hierarchical projection of relational data of the CDM. The XML View may use the document object model (DOM) as a universal representation for the XML View. The XML View typically maps entities to XML Elements and the relationships to the parent/child containment structure. The view is typically automatically integrated by the environment along a set of axes (such as containment and time), typically as a dynamic view, and is typically synchronized with the CDM in real-time. The XML View may provide an efficient process communication with the server by sending incremental updates between bulk updates. For example, if a user changes element E in the ~~Xml~~ XML element hierarchy of an XML View, the XML View class (or system) may determine which entity the element corresponds to. If the user has created a new relationship with the given entity, the

XML View / system may send an incremental update in the form of a message, which may indicate the kind of relationship created and entities connected by the relationship. If the user updates the data of an entity (e.g., ~~for example,~~ a field), the XML View / system may send an incremental update in the form of a message, indicating the entity to update, field name, and new field value. An embodiment of the XML View is explained more fully in U.S. Provisional Application No. 60/455,739 entitled "Network File System and Method", which is incorporated by reference herein, in its entirety, including the computer program listings of the appendix.

Please amend the paragraph at page 30 line 10 to page 31 line 2 as follows:

By way of example, Figure 9 is a functional block diagram showing an embodiment of compiling a unique view for the user. Figure 9 illustrates a folder 900, containing folders 915 and 920 through containment relationships 905 and 910. Access control 930 may consume through relationships 925 and 935 the folders 900 and 915. Access control 945 may consume through relationships 940, 950, ~~960~~ 955 the folders 900, 915, and 920. That is, access control 945 may grant access to folder 920 which is not granted by access control 930. The state may be the result of the following scenario. For example, User A, who may have access to folders 900 and 915 through access control ~~530~~ 930, may create ~~930~~ 940, creates derived access (perhaps by "attaching" folder 900 to an e-mail message) through access control 945 for user B. User A may also grant manage access to access control 945, user B. User B subsequently obtains an exclusive manage access (an exclusion on folder entity 900). User B then "inserts" folder 920 in folder 900 by way of the containment relationship 910. User A does not at that point gain access to the newly inserted folder 920, since user B holds an exclusion on folder 900. Therefore, when users A and B view the folder 900, the system may typically ~~generates~~ generate two separate views for each user. User A's view may show folder 900 as containing folder 915. User B's view may show folder 900 as containing folder 915 and folder 920. In this way, each user may receive a dynamic view based on their typically unique set of information contexts of which they are a member. When user B releases the exclusion, the system typically updates user A's view

in real-time based on the access evolution, which creates a new "consumer" relationship 960 connecting access control 930 and folder 920. In this way, user B may work "ahead" of user A in time (e.g., from user A's frame of reference) and B may sit "behind" user A in time (e.g., from user A's perspective), illustrating temporal continuity in the CDM, process of access evolution, and work flow.

Please amend the paragraph at page 31 lines 3 – 10 as follows:

Real-time updating provides an example of how implicit information (or managed information) may be consumed by an application (in this case, the file explorer). When the relationship 960 is created, the system typically responds to the change (in the form of relationship 960) by looking at all entities affected by the change. These entities may include folder 900. The system typically then notifies the application consuming folder 900 (i.e., both instances of file explorer, run by users A and B) of the change, or relationship 960. The application may respond by automatically updating the view to include ~~Folder~~ folder 920.

Please amend the paragraph at page 31 lines 11 – 14 as follows:

Figure 9 also provides an example of contextual information. As user B holds exclusive manage access of the ~~Folder~~ folder 900, user A may receive in the "context bar" information to the effect "user B is currently managing the selected folder" when folder 900 is selected.

Please amend the paragraph at page 31 lines 24 – 33 as follows:

Generally, the dynamic view may be considered a "2D" representation (typically flat, such as XML View) and mapping of "3D" relational data (typically the CDM). The dynamic view may be further considered by its typical process of real-time, bi-directional synchronization. The dynamic view may ~~be~~ further be considered by its typical ability to

reflect access evolution. The dynamic view may further be considered by its typical ability to link the information process (e.g., ~~for example,~~ including contextual information) to the representation (e.g., ~~for example,~~ allowing a user to navigate relationships attached to entities contained in the view). The dynamic view may further be considered by its typical ability to provide a standard "2D" view of relational data to an application.

Please amend the paragraph at page 32 lines 15 – 23 as follows:

The automation may receive several events, including "before change" and "after change". These events may belong to the general interface provided by the system as a component of the information process. The "before change" (or "pre change") event typically allows automation to respond to a change before it is made. The automation may be allowed to preempt the change by throwing an exception (which is typically captured by the platform). The "after change" (or "post change") event typically allows automation to respond to a change after it is made. The before and after change events may be a standard element of the platform API (e.g., ~~for example,~~ applications may process the same events).

Please amend the paragraph at page 32 lines 24 – 27 as follows:

The platform may "manage" automation as data persisted within the CDM. When managing automation, the system typically automatically loads and terminates automation based on data the automation may be known to consume (e.g., ~~for example,~~ through "consumer/access" relationships).

Please amend the paragraph at page 32 line 28 to page 33 line 15 as follows:

Figure 20 is a flow chart of an embodiment showing steps of in an instance of automation, starting at 2000. At step 2005, an entity 'E' consumed by automation 'A' may begin to change. For example, a user may have issued the system to create a relationship for which E is a target or have issued the system to update data belonging to the entity. At

step 2010, the system may automatically activate/raise automation A. At step 2015, automation A may receive information regarding a change in entity 'E' (e.g., ~~for example,~~ pre-change). At step 2020, automation A processes the proposed change. For example, the automation may analyze the proposed structure to maintain a constraint. At step 2025, automation A may or may not throw an exception (which may be captured by the system). An exception may indicate that the automation suggests the system may fail the change. For example, the automation may automate a constraint, which it may seek to enforce by throwing an exception. If 'yes', then at step 2030, the system may fail the update. At step 2035, the process ends. If 'no', then at step 2040, the system may allow the change (the system may also fail the change for other reasons). At step 2045, automation A may receive information regarding the changed entity 'E' (post-change). At step 2050, automation A may process the change (for example, checking a data structure) and may then signal the system that processing is "complete". At step 2055, automation A may make a change in response. For example, the automation may insert a new relationship in response, or may create information for the user who attempted to make the change (e.g., ~~for example,~~ providing notification). At step 2060, the system may automatically deactivate automation A. At step 2065, the process ends.

Please amend the paragraph at page 35 lines 7 – 19 as follows:

The context controlled by relationships includes, for example, a user's view of a given entity, control of the entity, and action taken upon the entity. Such control may include the evolving access model, which may give two users different views of the same entity (e.g., document) based on their role in a process of serial work flow. For example, the environment may display the draft version of a document to one user (e.g., who may be working on the latest set of best practices), and a previous version of the document to another user (who needs to review the working set of best practices). Another example of contextual control is the "activity", which contains a set of work/change contained across a set of files. The access constraints of the activity prevent any user who is not a member of the activity from seeing or editing document sections/changes belonging to the activity.

But, for those members of the activity, the environment may display work/change in real-time across the set of files, or allow the user to browse and review the set of change.

Please amend the paragraph at page 35 line 20 to page 36 line 7 as follows (Note this adds an indention to the start of the paragraph plus other changes):

_____The context provided by relationships may illustrate the meaning and appropriate interpretation of a given entity. For example, the activity previously illustrated also comprises a "semantic web" of association linking work/change belonging to the activity. As a result, a user who browses the content of a document containing such a change (assuming required access) is notified (e.g. in a side view) that the change was created as part of the activity, and for its stated purpose of the activity, in association with all other changes belonging to the activity (which the user may browse and review). Also associated with the change, and pushed into the users' side view, may include discussions and messages exchanged by users in the course of making the change, which allow the user to understand the exact wording of the change. If the user decides to edit the section (containing the change), the environment may automatically block other users from editing other members of the semantic association (protecting consistency of the association). The mechanism of collaboration – reconciling centralization and decentralization –may be analogous to the industrial revolution's assembly line, which continuously reconciled the division of labor among workers and individual stations (a decentralized process) and the integrated, collective product of their work (a centralized product). The invention may be considered as transferring collaborative complexity (typically managed in the minds of knowledge workers) into data and data relationships (within the managed CDM). Mitigating complexity by this mechanism may typically allow the invention to manage and enable collaboration on a large scale (e.g., ~~for example,~~ enterprise-wide or inter-enterprise collaboration and knowledge management).

Please amend the paragraph at page 36 line 32 to page 37 line 7 as follows:

[[|]]The invention embraces the e-mail paradigm as a powerful mechanism for creating collaborative context. The work flow model of the invention unifies e-mail and shared file management within a single, continuous information space. The invention may provide e-mail as an element of the network file system, reconstituting the infrastructure of e-mail within the CDM / platform. Messaging becomes an integrating factor (prior to the invention it is a "dis-integrating" factor), as users draw one another into shared information spaces, which belong to the continuous CDM. The access control model is able to allow a user to exchange e-mail in a way that is consistent with the current e-mail paradigm, while synchronizing and streamlining the subsequent process in real-time.

Please amend the paragraph at page 37 line 23 to page 38 line 8 as follows:

By way of example, Figure 24 is a flowchart of an embodiment showing steps of creating derived access through e-mail, starting at 2400. At step 2405, the user may select a set of entities to send via e-mail attachment. At step 2410, the user may instruct the system to send the selected files as e-mail attachment. At step 2415, the system may create an empty message entity. The user may alternatively send the message with no attachment. At step 2420, if the user made attachments, the system may insert the selected attachment entities as ~~a~~ children contained by the message entity. At step 2425, the system may present the e-mail editing interface, allowing the user to edit the body, recipient, and other fields of the message. At step 2430, the user may edit the message and other fields. At step 2435, the user may instruct the system to send the message. At step 2440, the system may create derived access for the message and other contained or attached entities (e.g., the set E). In creating derived access, the system may typically include in the access group the containment hierarchy of every entity belonging to the set E. For example, attaching a folder system F to message M may typically include all files contained by F (e.g., ~~for example,~~ the folder hierarchy) in the access group. At step 2445, the system may automatically ~~established~~ establish derived access for the message entity and nested entities for recipients of the message. The system may establish derived access

for recipients individually. The system may establish derived access for recipients as a group. At step 2450, the process ends.

Please amend the paragraph at page 38 lines 9 – 17 as follows:

The invention may include a hierarchical structure containing "organizational units" (OU's). The organizational unit typically contains other organizational units and users (in some implementations, the user may be considered an OU). The "group" is typically implemented as an organizational unit. Access control may designate the organizational unit as a recipient of shared access. For example, a group may receive shared access to an entity. Any member of the group may then receive access to the entity. The scope of an organizational unit in terms of security policy typically includes its membership (typically any user or organizational unit it contains carries access the OU derives).

Please amend the paragraph at page 38 lines 18 – 30 as follows:

By way of example, Figure 23 is a flowchart of an embodiment showing steps of creating derived access for a group of recipients, starting at 2300. At step 2305, the user may select a plurality of users. At step 2310, the user may assign the set of users access to one or multiple entities. At step 2315, the system may automatically take the set of users and create a group (a new organizational unit) containing each assigned user (group member). At step 2320, the system may automatically create derived access for the group. At step 2325, members of the group may automatically receive access to the entity(s) as a group. At step 2330, the process ends. Prior to the invention, setting up groups for purposes of collaboration typically involved ~~an~~ a central, administrative procedure (creating a static group). The invention may thus allow knowledge workers to self-organize in groups dynamically and easily perform group work (while preserving and remaining within a continuous and integrated data medium), as Figure 23 may illustrate.

Please amend the paragraph at page 39 lines 26 – 31 as follows:

The message may extend the file as a unified method of communication among workers. It may be a recursive structure enabling the ~~Message~~ message to become a threaded discussion among multiple workers, either synchronously (e.g., as an instant message discussion) or asynchronously (e.g., as an e-mail message). The user may insert an attachment (e.g., any file) within the body of the ~~Message~~ message. The application may display the attachment in-line and/or as belonging to a set of attachments.

Please amend the paragraph 39 lines 32 – 33 as follows (this includes language based on the originally disclosure at least page 41, lines 4-6):

The mechanism of "structured messaging" may thus be hierarchical. The mechanism is described in further detail for various elements of messaging, ~~including:~~ including email, instant messaging, and discussion threading, for example.

Please amend the paragraph at page 40 lines 1 – 12 as follows:

[[|]]The existing e-mail paradigm fits and is well behaved within the unified structure. The invention improves the e-mail paradigm by adding structure to a medium which is presently flat. Whereas the inbox is typically a list of messages (i.e., prior to the invention), the invention now includes the ability to create structure in a message store (such as My Inbox). The structure may be applied as, for example, [[:]] A "reply" (message A) to message B inserts message A as the child of message B, as though B were a folder containing a document. A "forward" (message C) of message A inserts message A as a child of message C. A file (F) "attached" to a message (D) inserts file F as a child of message D. The CDM allows and provides for a user to attach an entire folder system (since the root folder is file). The body of a message may also include "in-line" comments,

or messages which are related to sections of the body of the message. This allows a user to respond to sections of a message individually.

Please amend the paragraph at page 45 line 34 to page 46 line 13 as follows:

Figure 17A is a functional block diagram of an embodiment showing mixed serial/parallel work flow. In the example, the invention enables users to collaborate in any combination of serial and parallel work flow. In this example, a user U assigns two groups G1 and G2 a set of files contained by activity through access derivation from access control 1705. While both groups are working in the same activity file set, their work is separated by context, or as two information spaces. The first information space may be associated with access control 1710, corresponding to activity 1. The second information space may be associated with access control 1715, corresponding to activity 2. Work (change) done as part of activity 1 or 2 can then be accessed only by members of each activity, since access exclusion exists for each access control (1710 and 1715) assigned to groups G1 and G2. In this example, then, the activities provide multiple context in which group work is performed. Each activity may include a set of changes made by members, allowing members to navigate and consider the set of ~~change~~ changes as a whole (the sum group work effort). And each information space automates parallel workflow among members of an activity.

Please amend the paragraph at page 46 lines 14 - 21 as follows:

Referring again to Figure 4, the "dynamic view" provided to each user browsing or editing (consuming) the set of documents is constructed based on her activity membership (more specifically, the access control controls her granting access to the activity and member content). If the user belongs to only one of the activities, a document belonging to the file typically includes only change belonging to that activity. That is, the dynamic view may integrate all change belonging to activities of which the user is a member. For

example, if the user belongs to both activities 410 and 415, the dynamic typically integrates the sum change of each activity.

Please amend the paragraph at page 47 lines 16 – 21 as follows:

[[|]]The system may also provide "data reuse" through the temporal data medium. The system may implement data reuse in NETFS as a form of temporal containment. For example, a user may specify that a legal document contain another document at some time in the past. As a result, the system typically provides a view of the legal document embedding the "old" version of the contained document, even as other users may update the contained document.

Please amend the paragraph at page 48 lines 4 – 9 as follows:

The system may typically provide a file explorer application as part of NETFS, which may allow users to browse and manage all files within a single application (e.g., ~~for example,~~ documents, folders, messages, activities). The file explorer may provide a hierarchical view of files (illustrating the file containment structure). The file explorer may respond to the user's selection of a file by displaying a read-only view of the file in a pane or other window.

Please amend the paragraph at page 49 lines 1 – 8 as follows:

Figure 26 is a flow chart of an embodiment showing steps in the process of parallel work flow among users working in a shared document, starting at step 2600. At step 2610, the user may ~~opens~~ open document D in an application. In step 2615, the system may integrate the dynamic view of document D. In step 2620, the user may make a change to the document D. In step 2625, the system may persist the change and may send relational information to all user applications consuming document D which also have access to the

change. In step 2630, each consuming application may automatically update its data view to include the change. In step 2635, the process ends.

Please amend the paragraph at page 49 lines 16 – 25 as follows:

In a broader class, "hierarchy" imposed prior to this invention carries into many realms of computer data and systems management. The invention may free computer systems from this limitation. For example, the security model prior to the invention typically ~~require~~ requires that spaces fit within a tree. The result of the security model prior to the invention makes a lateral relationship impossible (or if a solution exists, it exists typically as a work around). For example, two companies would have extreme difficulty prior to the invention creating a shared data system (between servers regularly used to manage data in each organization, e.g., not a special purpose entity) during the course of an alliance, since neither security space may subsume the other. This holds for B2B and extranet relationships of many verities.

Please amend the paragraph at pages 49 line 26 to page 50 line 5 as follows:

The invention may ~~solves~~ solve the problem at a fundamental level by establishing what may be in part a *lateral* network dynamic. For example, two firms may easily and quickly establish a working collaborative relationship using the invention, since the data/information/security space may be considered continuous across each organization (when paired as a network). For example, a user in firm A may send an e-mail message to a user in partner/client B. The system typically creates a unique information context for that specific collaboration (with entity E), imposing no static hierarchy (at the level of the firm) in the security relationship between A and B, while allowing A to control B's access to entity E. In this sense, the context generated by an individual e-mail (or any greater collaborative construct) may be considered a dynamic virtual private network (VPN), which may enable powerful, lateral collaborative relations

between entities (e.g., ~~for example,~~ including enterprises and universities, or autonomous governmental agencies in the context of intelligence sharing).

Please amend the paragraph at page 50 line 31 to page 51 line 15 as follows:

The access model of the invention may enable an organization to control information exchange via the invention's security model and rules/policies/best-practices. The invention may allow an organization to define "information boundaries". Information boundaries may exist within a single organization or span multiple organizations. The information boundary may provide a container in which information exchange is limited. For example, a container may exist surrounding members of an organizational unit, or enclose a client and/or matter in the context of legal work (e.g., ~~for example,~~ including certain members of a law firm, a client enterprise, and an investment bank). As a part of the invention's file system, e-mail may be subject to additional security provided by an enterprise security implementation, such as, for example, Microsoft's ActiveDirectory. Since e-mail and files may remain in the single information space, users are not able to bypass security by moving data between systems. For example, a user may not be able to e-mail a file to a user who would not otherwise have file access. Prior to the invention, even if access to a file is restricted by the security model to a select group of users, a member can simply e-mail the file to unauthorized users. The invention may allow an organization to contain data within a single, secure space, while preserving the end-user experience (e.g., the user typically cannot unintentionally or maliciously move data out of the security context or boundaries).

Please amend the paragraph at page 51 line 27 to page 52 line 10 as follows:

Figure 19 is a flow chart of an embodiment showing steps of the information lifecycle, starting at step 1900. At step 1905, the user may extend derived access of entity E to a plurality of workers, beginning an information lifecycle. At step 1910, the users who received access may lose their access due to security policy. For example, a security policy

may allow only two weeks of access to the shared entity. Or, for example, a security policy may allow the users to retain access to the shared entity only while a business relationship exists between the firm of the user(s) extending access and user(s) receiving access (which may have ended). At step 1910, all users may lose access to the entity due to a file retention policy. For example, a file retention policy may automatically revoke access to data at a trigger point, such as three months after creation. At step 1920, the entity E may be automatically deleted from the CDM (e.g., ~~for example,~~ as ~~an~~ part of a file retention policy). At step 1925, the process ends. The entity may be cleanly removed from the system because the CDM typically prevents duplication of an entity (prior to the invention, files and messages scattered throughout user inboxes and hard drives make the enforcement of file retention policy nearly impossible). The invention may be considered as having a rope attached to an entity at all times, which it may retract at any moment (for security, information lifecycle, or other purposes).